L. 15.000

Supplemento a Commodore Computer Club N. 48 - Distribuzione MePe

SPECIALE UTILITIES

- DATA MAKER • SHUT OFF
- 4WD-RAMDISK • KOALA READER
- ASSEMBLER • DISK MERGER • TURBOKIT
- DISK MANAGER • AUTORUN MAKER • CRUNCHER
- TYPE • FORMAT SAVER • CROSS REFERENCE
- GRAPH COMPILER • RENUMBER • DELTAFIGHTER

in omaggio IL SUPERGAME DELTAFIGHTER

MENU

Per caricare il programma di presentazione è sufficiente digitare:

LOAD "*",8 (Return)

RUN (Return)

Sul video apparirà l'elenco dei titoli dei programmi presenti su questo disco.
Per caricare un determinato programma sarà sufficiente spostarsi con i tasti cursore e premere Return in corrispondenza del programma prescelto.
Per caricare i programmi indipendentemente dal Menu, si segurà la consueta procedura:

LOAD "PROGNAME",8 (Return)
RUN (Return)

TURBOKIT

E' una comoda utitlity che permette di velocizzare il caricamento dei programmi da disco di circa 5 volte.
Dapprima richiede il nome del programma caricatore (Loader) e successivamente il nome del programma oggetto (Object) che si desidera caricare in modo turbo.
Ad esempio, se abbiamo un programma di circa 16 Kbyte (32 blocks) che si chiama GRAPHICS che vogliamo caricare in modo turbo, sarà necessario rispondere con un qualsiasi nome alla domanda "LOADER NAME", ad esempio con "BOOT" e rispondere con "GRAPHICS" alla domanda "OBJECT FILE NAME".

Attenzione, il programma Object non deve avere lunghezza inferiore a 5 blocks per evitare malfunzionamenti.

DATA MAKER

Una utiliy che consente di trasformare una determinata zona di memoria in istruzioni DATA.
E' molto utile per chi debba trasformare i programmi in Assembler in istruzioni DATA per poi utilizzarli con programmi Basic.
La gestione avviene attraverso le finestre e la prima richiesta riguarda l'indirizzo di partenza della zona di memoria; è necessario rispondere con un numero compreso tra 6000 e 65535.
La seconda richiesta riguarda l'indirizzo di fine che deve logicamente essere compreso tra l'indirizzo di partenza e 65535.
Al termine della fase di input sarà richiesta una conferma, rispondere con Y o N, rispettivamente per si e per no.
Il tempo di elaborazione è direttamente proporzionale alla lunghezza della zona di memoria e, alla fine del lavoro, verranno listate le linee DATA.
E' ora possibile salvarle su disco con un normale SAVE, per ricaricarle è tassativo aggiungere il suffisso ",8" e non ",8,1" per un corretto caricamento.

ESEMPIO:

Supponiamo di dover trasformare in DATA la zona di memoria compresa tra 32768 e 33024 ($8000-$8100); dopo aver risposto con i valori di inizio e fine ed aver confermato inizierà la procedura.

Dopo un tempo di circa 12 secondi appariranno le istruzioni DATA, con numeri di linea da 1000 a 1036.

Il passo successivo è salvare i DATA, supponiamo con:

```
SAVE "DATAS",8
```

A questo punto bisogna spegnere e riaccendere il computer.

Per creare un programma Basic che allochi esattamente i numeri sarà opportuno operare come segue:

```
LOAD "DATAS",8
```

A questo punto il programma è in memoria e possiamo aggiungere le seguenti linee:

```
100 S=32768:REM START ADDRESS
110 READ A
120 IF A=-1 THEN END
130 POKE S,A
140 S=S+1
150 GOTO 110
.
.
.
10000 DATA -1
```

SHUT OFF

Una utility tipica di sistemi di elaborazione professionali; dopo circa 30 secondi di inattività della tastiera il video viene automaticamente spento.

Ciò va naturalmente a vantaggio della durata del video stesso nonchè di un minor consumo di energia elettrica.

Il video viene automaticamente riaccesoo non appena viene premuto un tasto qualsiasi.

AUTORUN MAKER

Per proteggere i propri programmi da occhi indiscreti, molto spesso è consigliabile dotarli di AUTORUN in modo che il caricamento avvenga con:

```
LOAD "PROGNAME",8,1
```

e si verifichi una esecuzione automatica senza la possibilità di essere fermato.

Autorun Maker fa questo lavoro per voi! E' necessario dare il nome del programma che si intende dotare di Autorun e il programma farà tutto da solo.

Il risultato sarà un programma che si chiamerà "PROGNAME-AR" e che sarà caricabile solo con ",8,1".

Attenzione, per maggior sicurezza è stata iinserita una protezione che non permette l'esecuzione del LIST.

CRUNCHER

Si tratta di un programma che permette di compattare qualsiasi tipo di file, programmi, testi, schermate grafiche in modo da risparmiare spazio sui dischetti.

Alla domanda "Source File Name", dovremo rispondere con il nome del file che intendiamo compattare e alla domanda "Object File Name" risponderemo con il nome che abbiamo deciso per il file compattato.

Alla fine della procedura di compattamento verranno evidenziate le lunghezze dei due programmi (Source e Object) ed il numero di blocchi su dischetto che sono stati risparmiati.

Per ricaricare un file compattato è sufficiente caricarlo normalmente quindi dare il Run per procedere alla procedura di "scompattamento".

Una volta effettuato lo scompattamento, il programma in memoria si presenterà come l'originale, per cui sarà possibile listarlo, modificarlo o mandarlo in esecuzione (sia con Run che con Sys).

La lunghezza dei programmi da compattare è limitata a circa 192 blocks e ciò e dovuto alla necessaria presenza in memoria di CRUNCHER.

L'eventuale spegnimento dello schermo durante l'esecuzione del programma segnala che l'operazione di "crunch" è stata interrotta a causa della eccessiva lunghezza del programma.

RAM-DISK

Tutti coloro che utilizzano computer evoluti, IBM, Atari, Amiga e così via, hanno certamente utilizzato una Ram-Disk.

La quasi totalità di utilizzatori di C/64-128, invece ignora quasi l'esistenza di questo potente strumento di lavoro.

Che cos'è una Ram-Disk?

Si tratta di un'area di memoria RAM che viene protetta da scrittura e che viene usata a tutti gli effetti come se fosse una periferica (virtualmente un disk-drive).

I vantaggi di questa partizione di memoria sono numerosi: in primo luogo, durante il lavoro di programmazione, non sarà necessario salvare continuamente su disco o nastro tutte le varie versioni del programma con notevole perdita di tempo, ma sarà possibile salvarle temporaneamente in memoria per poi salvare a fine lavoro un'unica versione su disco o nastro.

In secondo luogo, il programma è protetto da qualsiasi evento tranne il caricamento di un programma troppo lungo (superiore ai 110 blocchi).

Sia Run/stop e restore che il Reset non alterano il funzionamento del programma per cui il comando ***R (freccia a sinistra + R) è sempre presente e permette di passare dal Basic al modo Ram-Disk.

In terzo luogo funziona da "espansione": infatti, la memoria per il basic è limitata a $7FE0 (circa 30 KB) e lo spazio per la Ram-Disk è di circa 18 KB. In pratica con questa utility della 4WD-Soft si hanno 30+18 KB = 48 KByte (non male!)...

La Ram-Disk viene attivata con ←R (Return)
I comandi sono 9 e possono essere visualizzati con il comando HELP

- L "nome prg": carica da RAM il programma.
- S "nome prg": salva il programma nella zona RAM.
- V "nome prg": esegue la verifica del programma.
- D: elenca i programmi presenti indicandone anche l'occupazione in byte.
- F: formatta l'area di memoria cancellando gli eventuali programmi presenti.
- R "nome prg": rinomina il file in questione, dopo aver digitato ad esempio:

R "pippo" (Return)

verrà chiesto il nuovo nome.
- E "nome prg": cancella il programma in questione.
- H: elenca tutti i comandi.
- Q: esce al Basic.

Esempio d'uso

Supponiamo che abbiate registrato su disco o nastro un programma intitolato "Pax" e che vogliate modificarlo.

Per prima cosa sarà opportuno caricare e lanciare il programma Ram-Disk, successivamente sarà possibile caricare il programma "Pax" da disco o nastro e quindi digitare ***R (Return) e, una volta in ambiente Ram-Disk, dare il comando S "Pacchio".

A questo punto il programma "Pacchio" è in memoria (sia nell'area Basic sia nell'area Ram-Disk).

Tramite il comando Q sarà possibile tornare al Basic e modificare il listato originale e quando lo si ritiene opportuno salvarlo sotto Ram-Disk anzichè su periferica.

In questo modo si risparmierà una notevole quantità di tempo.

Una volta terminata la fase di modifica e/o miglioramento del programma sarà possibile salvarlo su disco o nastro.

Da calcoli approssimativi risulta che il numero dei salvataggi (e caricamenti) di un programma di media lunghezza prima della sua versione definitiva è di circa 80; tramite questa utility sarà possibile non solo evitare i "tempi morti" ma anche evitare un'usura eccessiva per la periferica.

Attenzione, lo spegnimento del computer comporta la perdita totale di ciò che è contenuto in RAM.

DISK MANAGER

Una potente utility per chi intenda lavorare con il drive.

Permette di modificare i nomi dei file, cancellarli, recuperarli e proteggerli o sproteggerli dalla cancellazione.

Le opzioni sono le seguenti:

1/ UNSCRATCH, permette di recuperare un file cancellato.
2/ SCRATCH, cancella un file.
3/ LOCK, protegge un file dalla cancellazione.
4/ UNLOCK, toglie la protezione di cui sopra.

Durante la fase di display vengono visualizzate, accanto al nome del file il tipo di file (PRG,SEQ,USR,REL,DEL), il numero di blocchi occupati e la traccia e settore in cui parte.

TYPE

E' un comando di ambiente MS-DOS, molto comodo che permette di esaminare il contenuto di file di testo (SEQ).

Premendo SHIFT si ferma la visualizzazione mentre premendo Run/Stop si termina.

CROSS REFERENCE

Per il debugging dei programmi è necessario sapere in quali linee di programma vengono aggiornate le variabili.

Con

SYS 49152, X

dove X e il numero di colonne con cui vogliamo l'output (12-255), verranno indicate tutti i nomi delle variabili del programma Basic in memoria con accanto i numeri di linea in cui vengono utilizzate.

E' possibile dirigere l'output su stampante con:

OPEN 4,4:CMD4:SYS49152,X

DISK MERGER

Capita spesso di dover fondere due programmi Basic in uno solo.
Purtroppo il C/64 non possiede questo comando ma, tramite questo programma sarà possibile fondere due programmi presenti sullo stesso disco in un terzo che pure verrà salvato sul medesimo disco.

Esempio:

Supponiamo di voler fondere i seguenti programmi, presenti su disco con i nomi PROG #1 e PROG #2 e di volerli fondere in un terzo di nome PROG #3.

```
10 REM *** PROG #1 ***
15 PRINT "I AM THE FIRST"
50 PRINT "PROGRAM"
```

```
20 REM *** PROG #2 ***
30 PRINT "I AM THE SECOND"
70 PRINT "OF THE TWO PROGRAMS"
```

Dopo aver mandato in esecuzione il programma MERGER, il risultato sarà questo:

```
10 REM *** PROG #1 ***
15 PRINT "I AM THE FIRST"
20 REM *** PROG #2 ***
30 PRINT "I AM THE SECOND"
50 PRINT "PROGRAM"
70 PRINT "OF THE TWO PROGRAMS"
```

RENUMBER

Programmando capita di dover creare spazio tra due linee Basic per inserire una o più nuove istruzioni.
Molto spesso però se la numerazione non lo consente è necessario ricorrere ad utility di Renumber che rinumerano il programma.
La maggior parte di queste utility però non aggiornano i GOTO e i GOSUB con conseguenti malfunzionamenti.
La nostra routine di Renumber non ha invece questo inconveniente.
Le istruzioni per l'uso compaiono sul video alla partenza del programma.

KOALA READER

Molto comodo per chi lavora con il Koala (un diffuso programma grafico) in quanto consente di visualizzare le immagini senza caricare Koala.
Se si risponde con "$" verrà visualizzata la directory del dischetto.

ASSEMBLER MASTER

E' formato in realtà da tre programmi monitor per linguaggio macchina allocati in diverse zone di memoria.

Il primo è allocato a 4096 ($1000), il secondo a 32768 ($8000) ed il terzo a 49152 ($C000).

I comandi principali sono i seguenti:

A: permette di assemblare da una zona di memoria, es.

A 2000 lda #$00.

D: disassembla da una locazione,
es. D 2000.

F: riempie una zona di memoria con un certo valore,

es. F 2000 3000 EA

riempie la memoria da $2000 a $3000 con il byte EA.

G: esegue la routine all'indirizzo specificato,

es. G 2000, esegue la routine allocata a $2000 (8192).

H: ricerca il byte o la stringa specificata in una zona di memoria.

es. H 2000 3000 A9 00

ricerca nella zona compresa tra $2000 e $3000 la sequenza A9 00; se si sostituisce alla sequenza una stringa di caratteri, ad esempio 'COMPUTER, la ricerca avverrà sulla stringa.

I: ispeziona la memoria,

es. I 2000 3000

mostra il contenuto da $2000 a $3000.

T: trasferisce da una zona di memoria,

es. T 2000 3000 4000

trasferisce la zona da $2000 a $3000 in $4000.

P: dirige l'output su stampante, qualsiasi comando verrà diretto alla stampante.

O: chiude la comunicazione con la stampante.

L: carica un programma in assembler,

es. L "PROGNAME",08

S: salva un programma in assembler,

es. S "PROGNAME",08,2000,3001, salva un programma da $2000 a $3000.

La directory del disco è visualizzabile con il seguente comando:

]$

Si raccomanda per i principianti una lettura di materiale riguardante l'assembler del C/64.

FORMAT SAVER

Nella malaugurata ipotesi in cui abbiate formattato involontariamente un disco, contenente dei files, questa utility vi permetterà di recuperarne il contenuto, ripristinando la directory.
Il programma funziona solo in caso di formattazione veloce, quella senza 'ID', es: OPEN 1,8,15,"N0:DISK NAME"
Una volta attivo, il programma controlla se effettivamente il disco è stato formattato, al fine di evitare di alterare dischi contenenti dei dati utili.
Successivamente, vengono recuperati gli eventuali files cancellati, e per ognuno di questi, viene inserito come nome, nella Directory, un numero progressivo in quanto è impossibile reperire il nome originario.
Sarà vostra cura caricare i vari files recuperati, analizzarne il contenuto e assegnare loro il nome più appropriato con il comando RENAME, es: OPEN 1,8,15,"R0:NEW NAME=OLD NAME".

GRAPH COMPILER

L'idea da cui è nato il Graph Compiler era quella di costruire un prodotto di nuova concezione.
In più si voleva che facesse l'impossibile: cioè che compilasse anche istruzioni non contenute nel veramente povero linguaggio basic della macchina C64 (come i comandi DRAW e PLOT).
In un certo senso il Graph Compiler riesce a fare ancora di più: esso infatti "compila" pure istruzioni che sono di notevole utilità e che sono assenti anche nei linguaggi basic più evoluti come quello del C128; infatti si possono "compilare" istruzioni completamente inedite (INIT e C-DRAW) che verranno descritte più avanti e che permettono di ottenere abbastanza facilmente schermate di grafici tridimensionali con linee nascoste.

Non ci si può aspettare miracoli da un compilatore grafico-matematico. Sono oramai molto diffusi i cosiddetti basic estesi che già traducono in linguaggio macchina i comandi grafici. D'altra parte, un compilatore (per ovvie ragioni di dimensioni) dovrà utilizzare le lente routine dell'interprete basic per calcolare (ad esempio) sin(x). Dunque il calcolo di sin(x) e, più in generale, il computo di espressioni matematiche sarà lento sia in basic che in linguaggio macchina.

Ciò nonostante, l'aumento di velocità che si ottiene con il Graph Compiler è veramente cospicuo.
E' però nella grafica tridimensionale che si ottengono i migliori risultati.

Un altro vantaggio del Graph Compiler è che il programma oggetto può essere posizionato con una certa libertà nella memoria del C64 (ciò non è permesso dalla maggior parte dei compilatori). Esso inoltre accetta come periferica sia il drive che il registratore.
Dobbiamo infine avvertire che non tutte le istruzioni BASIC del C64 sono compilabili (il programma sarebbe diventato enorme), ma solo quelle che hanno un interesse grafico-matematico.
Vediamo ora con maggiore dettaglio il programma.

Istruzioni compilabili

Comandi basic usuali

END,LET,GOTO,IF,GOSUB,RETURN,REM,POKE,PEEK

Sono inoltre compilabili i seguenti comandi Basic usuali, ma con una sintassi leggermente diversa:

WAIT

Non richiede parametri.In corrispondenza a tale comando, l'esecuzione si arresta fino a quando viene battuto un tasto.

SYS

Non è ammessa la SYS calcolata;.vale a dire una riga di programma del tipo:
100 A=49152:SYSA
deve essere sostituita dalla seguente:
100 SYS49152

THEN

Non sono permesse le abbreviazioni. Cioè a dire la riga
100 IFA=1THEN200
deve essere sostituita dalla seguente
100 IFA=1THENGOTO200

DEF @

La sintassi è la seguente (N=0,1,...15):

DEF @ N=espressione matematica di variabili reali.

Esempio di schema programma con il comando DEF:

```
100 DEF @ 4=EXP(-1/(X+Y+Z))
........................
500 X=14:Y=-25:Z=27
600 A=1+2* @ 4
700 END
```

Alla riga 600 il programma, a partire dai valori attuali di X,Y,Z, si calcola # #4 e poi ricava A.

Nella definizione di una funzione non si può utilizzare (direttamente o indirettamente) il simbolo della funzione stessa; non si può utilizzare cioè la ricorsività.Ad esempio una riga di programma del tipo:

100 DEF @ 4=1+@ 4

è severamente proibita, in quanto il programma , per calcolare @ 4 dovrebbe calcolarsi @ 4 stessa, ma ciò richiederebbe il calcolo di @ 4 stessa e così via. Il Graph Compiler abortirebbe dando il seguente messaggio:

SYNTAX ERROR IN 100

Per ragioni analoghe è altrettanto proibita una riga del tipo:

100 DEF@4=1+@3:DEF@3=X-@ 4

In tale caso il Graph Compiler non si accorge dell'errore. Mandando, però, in esecuzione il programma oggetto, il computer si blocca.

Istruzioni matematiche

Sono compilabili le seguenti operazioni:

+,-,*,/, ↑,AND,OR

e le seguenti funzioni:

I gruppo (funzioni usuali)

SGN,INT,ABS,SQR,LOG,EXP,COS,SIN,TAN,ATN

II gruppo (funzioni non presenti nel basic del C64)

ASN=arco seno (-1<x<1)
ACS=arco coseno (-1<x<1)
SNH=seno iperbolico
CSH=coseno iperbolico
TNH=tangente iperbolica
ASH=arco seno iperbolico
ACH=arco coseno iperbolico (1<=x)
ATH=arco tangente iperbolica (-1<x<1)
PP=parte positiva (cioè PP(x)=(ABS(x)+x)/2)
PN=parte negativa (cioè PN(x)=(ABS(x)-x)/2)

Nell'elenco che sopra abbiamo fatto di tali funzioni, sulla destra del nome, si può trovare, fra parentesi, l'intervallo su cui sono definite. Ciò vuol dire che di fronte a una riga di programma del tipo:

100 A=ASN(10)

il compilatore non farebbe obiezioni, però, mandando in esecuzione il programma compilato il computer risponderebbe con un "ILLEGAL QUANTITY ERROR". Questo fatto non dovrebbe risultare strano a un programmatore esperto il quale sa bene che già nel basic usuale una riga del tipo:

100 A=SQR(-1)

non verrebbe accettata dal computer propio perchè -1 non appartiene all'intervallo di definizione della funzione radice quadrata.

Quando l'intervallo di definizione non compare, vuol dire che non ci sono problemi, cioè che la funzione si può calcolare per ogni valore senza alcuna limitazione.

Istruzioni grafiche

Sono presenti le seguenti istruzioni grafiche aggiuntive:

GRAF

Sintassi: GRAFN,M
N e M devono essere numeri compresi fra 0 e 15. Attiva il modo alta risoluzione dello schermo ponendo i pixel accesi del colore di codice N e i pixel spenti del colore di codice M.

NORM

Si usa senza parametri; fa ritornare lo schermo nel modo risoluzione normale.

CLEAR

Si usa senza parametri. Inizializza tutta la pagina grafica, spegnendo tutti i pixel dello schermo.

PLOT

Sintassi: PLOTX,Y

X e Y possono essere numeri o espressioni numeriche. Accende il pixel di coordinate X,Y. L'origine è posta al centro dello schermo. Se il punto da accendere dovesse (idealmente) trovarsi fuori dallo schermo, le coordinate verrebbero automaticamente troncate e verrebbe acceso un punto della cornice.

UNPLOT

Comando identico a PLOT, che però spegne il pixel invece di accenderlo.

DRAW

Sintassi: DRAWX,Y,W,Z

Tale comando accende i pixel del segmento avente estremi (X,Y) e (Z,W). Gli altri dettagli sono analoghi a quelli del comando PLOT.

INIT

Si usa senza parametri. Inizializza lo schermo ad alta risoluzione nel tracciamento di grafici tridimensionali ad alta risoluzione con righe nascoste. Vedi anche la descrizione del comando CDRAW.

CDRAW

Comando con sintassi identica a DRAW; i pixel, però, vengono accesi solo se è verificata una condizione che è inizializzata dal comando INIT. Quando il programma esegue il comando CDRAW, vengono accesi i pixel solo se il valore della seconda coordinata è esterno a un certo intervallo. I grafici devono essere disegnati accendendo prima i punti che sono (in prospettiva) più vicini all'osservatore. Per altri dettagli di tale comando vedere più avanti la descrizione dei programmi dimostrativi 3D, 3E e 3F.

Uso pratico del Graph Compiler

L'uso del Graph Compiler è molto semplice: spegnete ed accendete il C64, caricate il Graph Compiler, mandatelo in esecuzione e seguite le istruzioni sul video che dovrebbero essere tutte autoesplicative.

Durante la compilazione dovrete fornire il nome del programma sorgente (supponiamo che

tale nome sia PROG1) e l'indirizzo di partenza della compilazione (supponiamo che sia 49152).

Alla fine il programma compilato si troverà in memoria e sarà automaticamente salvato con il nome:

+PROG1.....49152

Per mandare in esecuzione il programma oggetto basterà dare il comando (in modo diretto o in modo programma):

SYS IND

dove IND è l'indirizzo di partenza della compilazione. Nell'esempio considerato sopra dovremo dare il comando SYS 49152.

Alla fine della compilazione, viene inoltre visualizzata l'area di memoria occupata dal programma oggetto.

Se vengono usati i comandi grafici, saranno ovviamente occupate altre zone di memoria per la pagina grafica, la memoria colore e la inizializzazione dello schermo. Tali zone sono però state sistemate nella cosiddetta RAM nascosta del C64, cioè sotto banchi ROM. La RAM usuale, non esplicitamente occupata, è dunque tutta a disposizione del programmatore per altri programmi BASIC o in linguaggio macchina.

I programmi oggetto così ottenuti possono essere caricati in memoria con il comando:

LOAD "NOME",8,1

Avvertenze importanti

A) Il programma sorgente dovrà concludere la propria esecuzione con il comando END.

B) Il Graph Compiler riconosce solo le variabili reali: dunque non compila variabili del tipo A% o A$. Non sono peraltro ammesse le variabili vettoriali: non viene dunque accettata ad esempio l'istruzione A(5)=1.

C) L'istruzione FOR...TO...NEXT non è compilabile. Ciò può sembrare una grossa limitazione. Si può, però, osservare che ogni ciclo FOR...TO...NEXT, può essere sostituita da un ciclo IF...THEN...GOTO. Facciamo un esempio. Il seguente schema di programma:

```
100 FORI=1TO10
110 ..........
...............
200 NEXTI
```

può essere sostituito dal seguente:

```
100 I=1
110 ..........
...............
200 I=I+1:IFI<=10THENGOTO110
```

D) I programmi oggetto usano per memorizzare le variabili le routine dell'interprete BASIC. Ciò significa che le eventuali variabili libere possono essere modificate con il linguaggio BASIC. E' questa un'altra grossa differenza rispetto ai compilatori commerciali, che risulta molto utile, come si vedrà più avanti.

I programmi dimostrativi

Troverete, accanto al compilatore grafico-matematico, anche alcuni programmi che sono

scritti nel linguaggio compilabile dal Graph Compiler (sono cioè programmi sorgente). Non mandateli dunque in esecuzione con un RUN perchè il C64, non interpretando i nuovi comandi, risponderebbe con un antipatico SYNTAX ERROR. Tali programmi devono essere invece compilati.

Ora non vi resta altro che provare la funzionalità del Graph Compiler; l'utility che non può mancare nella vostra libreria software.

DELTAFIGHTER

In questo avvincente gioco vi troverete alla guida di un caccia a reazione, armato di missili: la vostra missione è quella di attraversare tutto il sofisticato sistema di difesa dei vostri nemici, addentrandovi il più possibile nel loro territorio, mentre avversari sempre più agguerriti tenteranno di fermarvi.

Dai primi elicotteri in lento volo, incontrati sul mare ancor prima della costa, fino ai velocissimi jet che compaiono molto più avanti, tutto vi sarà ostile,con un livello di difficoltà che cresce mano a mano che il vostro Deltafghter si addentra nelle difese nemiche.

Certamente solo un bravo pilota può portare questa missione a compimento: spetta a voi dimostrare di esserlo, raggiungendo l'aeroporto nemico, tutte le installazioni che seguono, fino a conquistare il Bunker Corazzato!

Il vostro jet è guidato dal joystick in porta 2; durante il gioco vengono segnalati sul bordo inferiore dello schermo i punti totalizzati e le "vite" rimaste. Queste ultime sono tre all'inizio del gioco, con un nuovo "bonus" al raggiungimento dei 10000 punti.

Viale Famagosta 75
tel. (02) 8467348/9/40
20142 Milano